DEPARTMENT OF DEFENCE

AR-001-472

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION

ELECTRONICS RESEARCH LABORATORY

TECHNICAL REPORT

ERL-B123-TR

JINDALEE PAPER NO. 123

COMPUTER CONTROL OF THE JINDALEE STAGE A HF TRANSMITTING STATION

A.M. Forbes

S U M M A R Y

Between 1976 and 1978, the JINDALEE experimental over-the-horizon radar in Central Australia has had an HF radio transmitting station controlled by a PDP11/10 minicomputer, using a specially developed program written in the MACRO-11 assembly language. This paper shows how the design and maintenance of the program were influenced by the available facilities, and goes on to describe the computer control that was developed. A recommendation is made that computer installations at remote sites should be self-sufficient in programming aids.

POSTAL ADDRESS: Chief Superintendent, Electronics Research Laboratory,
Box 2151, G.P.O., Adelaide, South Australia, 5001.

TABLE OF CONTENTS

LIST OF APPENDICES

LIST OF FIGURES

1. INTRODUCTION

One of the functions of the JINDALEE experimental over-the horizon radar has been to transmit high frequency radio signals from central Australia, concentrating the radiation towards the north-west, and to analyse received signals backscattered from the earth's surface in the target region of interest. Between 1976 and 1978, this radar was operating at a receiving station at Mt. Everard, along the Yuendumu Road near Alice Springs, while the corresponding transmitting station was near Harts Range on the Plenty Highway, some 100 km across country from the receivers. The 100 km spacing was planned to minimise the chance of contamination of the received backscattered signals.

It was decided that, in view of the complexity of the experiments, the transmitting station should be under the control of a minicomputer, responding to parameter settings sent over a communication link from the Receiver Site. This paper describes the organisation of the computer program written for the control of the transmitting station. The description includes how the program was used by the Transmitter Site staff, and its interaction with computer programs at the Receiver Site.

2. EQUIPMENT CONTROL REQUIRED

2.1 Radar

Perhaps the most noticeable feature at the Transmitter Site is the antenna field, which includes an array of 16 vertical log-periodic antennas used by the radar. For Stage A, the radar array was divided into 4 sub-arrays of 4 antennas each, each subarray being driven by one of 4 HF power amplifiers. The low-level radio frequency drive for the power amplifiers was obtained from an arrangement of frequency synthesisers and translators, programmed to produce the necessary radar modulation. Details of the radar format are beyond the scope of this paper.

The control functions required to operate the radar transmitters included:

(a) establishing and maintaining precise timing, corresponding to the time kept at the Receiver Site;

(b) setting up radar parameters, e.g. frequency and bandwidth;

(c) tuning the power amplifiers for the operating frequency, which was done with an electro-mechanical servo system;

(d) keying the power amplifiers on and off;

(e) dealing with amplifier fault conditions;

(f) controlling the radiated power level; and

(g) changing beam direction by time-delay steering.

2.2 Backscatter Sounder

The sounder equipment was used for obtaining backscatter ionograms, required for choosing operating conditions for the radar. At the Transmitter Site this equipment consisted of a linear FM-CW (Frequency Modulated, Continuous Wave) sweep generator, consisting of a digital controller that swept the output of a frequency synthesiser upwards at 33.3 kHz/s within specified limits of the HF range. The synthesiser output was amplified by a fifth power amplifier, whose output was fed to a 17'th log periodic antenna installed near the radar array. At the Receiver Site, another FM-CW linear sweep generator served as the local oscillator for the sounder receiver, down-translating the back-scattered signals for spectral analysis.

The control functions required for operating the sounder transmission included:

(a) keeping the Transmitter Site sweep in step with that at the Receiver Site;

(b) setting up the lower and upper frequency limits for the sounder sweep;

(c) pre-tuning the power amplifier to a fixed frequency at the bottom of the region to be swept, necessary to enable the amplifier to follow the swept frequency;

(d) allowing the power amplifier to tune and load into its antenna continuously, adjusting to the upwards-sweeping operating frequency;

(e) keying the power amplifier on and off; and

(f) dealing with power amplifier fault conditions.

2.3 Telecommanding of remote site

There was equipment installed at Derby, on the north coast of W.A., that needed to be remotely controlled during the course of the JINDALEE experiments. The telecommand signals used a computer-controlled modulation generator that worked with the same frequency synthesiser used for sounding. If a telecommand operation was needed while the sounder was radiating, control of the synthesiser was transferred from the sounder Linear Sweep Controller to a digital source that set up the telecommand frequency, while the Linear Sweep Controller continued operating. When telecommanding was finished, the synthesiser would be again controlled by the Linear Sweep Controller. The same power amplifier and antenna were used as for the sounder, and power amplifier control was similar.

2.4 Backup operation

It was possible for the four radar power amplifiers and their associated equipment to be used for backscatter sounding or telecommanding, which might be done if some of the normal sounder/telecommand equipment was unservicable, or if the larger signal strength from the 4 amplifiers and 16 antennas, instead of the normal 1 amplifier and 1 antenna, was needed. In this mode, called the Backup mode, radar operations had to be suspended.

3. COMPUTING SYSTEM HARDWARE

3.1 Transmitter Site

The minicomputer provided for Transmitter Site control was a DEC PDP11/10, with 16k words of core storage and a DECwriter console terminal. Equipment was controlled through a series of asynchronous byte-oriented I/O modules connected to the UNIBUS, so that if the program moved data to or from the appropriate UNIBUS address, a bit pattern would be transferred across an equipment I/O interface. Certain radar parameters were implemented through a specially designed timing card plugged into the UNIBUS. Communication with the Receiver Site was through a radio Data Link, used firstly for loading the program, then for sending control parameters to the transmitter system. For experimental data logging, to be described later, a low data rate A/D converter with an 8 channel analogue input multiplexer was included, interfaced to the computer through a DR11-C unit. The combination of multiplexer, conversion device and interface will be referred to in this paper as the 'A/D converter'.

A description of the PDP11 computer and its standard peripherals can be found in references 1 and 2, and the hardware configuration used at the Transmitter Site is shown in figure 1.

There was no provision for external data storage at the Transmitter Site. The computer installation, therefore, could not be free-standing, but always had to rely on other systems for program loading and any other software support.

3.2 Data Link to Receiver Site

The Data Link operated in full duplex mode between DL11 asynchronous serial interfaces, with a data rate of 244 bits/s., or about 24 bytes/s. The link equipment in each direction used bi-phase modulation of a VHF carrier, whose frequency was of the order of 70 MHz. The low data rate was required to give the Data Link receivers a narrow noise bandwidth, allowing for signal fading along the path between the sites. The Data Link worked very reliably, with total down-time due to VHF signal fading of the order of a minute in 24 h. As a fade would rarely last more than a few seconds, the down-time was not noticeable operationally.

3.3 Supporting systems

The computing system at the Receiver Site had two PDP11 machines, referred to respectively as the Radar Computer and the Data Logger Computer, each running under an RSX11-D operating system. Transmitter Site support from these systems consisted of program storage on disk and magnetic tape, and text editing, language translation, and down-line loading facilities.

At Salisbury, the PDP11 Backup Computer, also running under RSX11-D, was used for program development and maintenance. The IBM370 Central Computer was useful for source module storage and editing, for producing large line-printer listings, and for some of the program modelling as described later in paragraph 5.4.

4. CHOICE OF A LANGUAGE

The Transmitter Site control program had to be written in a language which allowed it to be translated into a load module for down-line loading over the Data Link, preferably using the same absolute binary format as the PDP11 paper tape system. Among the requirements for the program and its language were the following:

(a) it had to be compatible with an RSX11-D host system;

(b) the resulting load module had to be small enough to fit into the core storage available at the Transmitter Site;

(c) at least 24 bit numeric precision was needed for internal arithmetic, to allow frequencies to be expressed to 1 part in 10**8;

(d) versatility was needed with respect to numeric formats, for handling binary and binary-coded-decimal data through the I/O modules; and

(e) a communication protocol was needed for working with the Receiver Site programs.

As a generalisation, it can be stated that computer programs can be written in low-level or high-level programming languages.

The writing of a program in a low-level language nearly always requires coding in considerable detail, as the program statements have to control the internal features of the machine. Such a program is difficult to understand unless the person studying it is familiar with the computer and its set of machine instructions.

A programmer using a high-level language does not need to concern himself with much of the internal operation of arithmetic, control, I/O, etc., but can concentrate on producing a program to solve his particular problem. Because a program written in such a language is relatively easy to code, read, understand, and modify, any computer program should be written in a high-level language whenever practical. Even if a high-level language cannot be used, it is still desirable to use as many pre-defined functions as possible from available program libraries and other sources.

The following software packages and uses of high-level programming languages were considered, but as it will be seen, none of them proved suitable or sufficiently versatile.

(a) RSX11-S, the stand-alone version of RSX, would have been ideal, as it allows tasks to be coded in FORTRAN and assembly language and then linked and down-line loaded. Unfortunately, at the time the program was being written, RSX11-D could not be used as a host system, but only RSX11-M.

(b) The RSX task builder, the only linkage editor available, produces task images that are not suitable for down-line loading into a stand-alone computer; therefore the program could not be linked together from separately compiled or assembled modules.

(c) Stand-alone BASIC and FOCAL language interpreters were available, which could be down-line loaded; but they involved inefficient coding in the high level language, and extensive enhancements in assembly language.

(d) The PDP11 paper tape software packages IOX (I/O Executive) and FPMP-11 (Floating-Point Math Package) offered some facilities for I/O and arithmetic operations, but not sufficient to warrant their modification and use.

However, the MACRO-11 assembler supplied with the RSX operating system can produce an absolute binary machine-code output, with a format similar to that used by the paper tape systems. MACRO-11 proved to be the only language that could be used; although every part of the program had to be coded in detail, because there was no linkage editor to give access to libraries of standard system routines, and the RSX system directive macros were not appropriate.

The PDP11 machine instruction set is described in reference 1, and the MACRO-11 assembly language is described in reference 3.

5. PROGRAM DEVELOPMENT AND MAINTENANCE

5.1 Editing

When all the updates had finally been made, the program's source code was some 9000 lines long. This occupied 400 blocks of storage out of the 4800 available on the RK05 disk pack used by the RSX systems at Salisbury and the Receiver Site. During source text editing with the EDI utility program, a work file is created with the same size as the source file, so that editing of the Transmitter Control Program under RSX engaged some 800 blocks, a considerable portion of the disk. At the Receiver Site, most source program files were stored on special 'source' or 'diagnostic' disks. When a program change was needed, data gathering was stopped, any special disks were mounted, and a program updating session was conducted. There were seldom any conflicting requirements for use of the computing system, and there was usually ample disk space available for program development. However, at Salisbury, where most editing changes were done on the Transmitter Control Program, prior to debugging and installation at the Transmitter Site, the Backup RSX system was used in a multi-terminal, multi-user configuration. Editing the Transmitter Control Program on the Backup computer was difficult, because the disks were normally almost full, and difficulties were often encountered in getting enough task priority or memory for running the editor.

Therefore, editing of the Transmitter Control Program at Salisbury was normally done on the IBM370 Central Computer, using the TSO (Time Sharing Option) text editor. Program files were transferred to the IBM system on 800 bpi magnetic tapes, using a program running on the IBM machine for reading RSX FILES-11 tapes into datasets for TSO editing. After editing, new FILES-11 tapes were created for transferring the source files back to the RSX system. Tape record blocking and deblocking and label generation had to be done explicitly, as the IBM system software did not fully support the version of the ANSI D tape format used by RSX.

5.2 Assembly

The output obtained from the assembly process depended on the switch options used with the MACRO-11 assembler. The output might include absolute binary code occupying about 40 blocks on an RK05 disk, and/or a complete assembly listing of about 800 blocks, and/or a symbol table of some 30 blocks.

The Absolute Binary output from the assembler was stored in a disk file, in the variable-length record format shown in Appendix I.

When an assembly was done at Salisbury on the PDP11 Backup computer, the source file was normally read from a magnetic tape volume, either generated using the IBM370 Central Computer as described above, or sent back from the Receiver Site as a copy of a working program. The complete assembly listing was written to a second magnetic tape, which was then taken to the Central Computer, for running of a program that produced the 9000 line, 220 page listing on a line printer. Measures were taken to prevent the Backup system from printing the full listing, which would otherwise have put too heavy a load on its facilities. Normally there was no need at Salisbury for the absolute binary output, as the Transmitter Control Program could be run on only the one machine, at the Transmitter Site.

At the Receiver Site, the source file was normally stored on an RK05 disk, having been copied from a tape sent from Salisbury and in some cases subsequently edited during development at the Transmitter Site. The binary output from assembly was stored on disk, for further processing to produce the absolute binary stream used for down-line loading, as described later. As a general rule, program listing was confined to the assembly symbol table, to help with debugging at the Transmitter Site. The complete assembly listing was usually too expensive in system resources to be printed at the Receiver Site, but was produced instead at Salisbury from a magnetic tape, and then sent back to Alice Springs.

5.3 Down-line Loading

The PDP11/10 at the Transmitter Site was equipped with a hardware bootstrap loader, which was used for loading an absolute binary loader over the Data Link from the Receiver Site. This was needed only when the Absolute Loader was unuseable from having been accidentally overwritten. The Absolute Loader was run to load in the Transmitter Control Program, which started running as soon as its loading was finished. The foregoing represents normal PDP11 paper tape loading practice, except that the data to be loaded came from a communications link instead of from a paper tape reader. An HF voice radio link was used to co-ordinate the loading operations at the Receiver and Transmitter Sites.

As it was considered desirable, for systems compatibility, to use the standard DEC paper tape absolute binary format for down-line loading, some extra information had to be added to each binary record generated by the assembler. The details are listed in Appendix I.

The PDP11 Absolute Binary Loader, a standard software product distributed by DEC with their Paper Tape Software packages, did not prove suitable for loading data over the Data Link. When that loader program detects a checksum error it halts the computer and with it the input stream from the tape reader, whereupon standard operator action is to backspace the paper tape, and to press the CONTINUE key on the computer Operator's Console to proceed. An equivalent procedure might conceivably have been followed when the data stream came from the Data Link: should the loader program have halted because of a transmission error, a request might have been sent to the operator at the Receiver Site to stop the data stream and backspace the file; an impracticable procedure. Furthermore, the loader does not detect checksum errors until the data has already been loaded into memory, so that a corrupted load address or field length caused by a transmission error can have unpredictable results.

Therefore a special Absolute Loader was written for this application. It stored the data in a temporary buffer within the loader program, and did not transfer it to its destination in main memory until the checksum had been verified. Error-free receipt of a data record was then acknowledged by sending back the load address to the Receiver Site. If there had been a checksum error, the data was neither loaded nor acknowledged.

The Absolute Loader program, written in the PDP11 paper tape bootstrap format, was sent over the Data Link to the Transmitter Control Computer by an RSX task called 'TXBOOT'. When this task was initiated, it sent the bootstrap leader code of octal '351' bytes over the link as a continuous stream. At the Transmitter Site the hardware bootstrap loader was started from the switches on the Operator's Console, then at the Receiver Site the appropriate keyboard command was made to TXBOOT to make it send the absolute loader code. If loading was successful, the Transmitter Site Control Computer would halt with the start address of the absolute loader in the Operator's Console display. If loading was not successful, which sometimes happened, presumably because of transmission errors, the process had to be repeated. Sending of the absolute loader took about 10 s transmission time.

The down-line loader program at the Receiver Site used a double-buffering technique for reading the binary load module file. Each record to be sent over the Data Link was stored in one of two buffers, and it would not be replaced by a new record from the disk file until its load address had been received back. In this way, any record not received correctly was automatically repeated. It took about 25 min to load the Transmitter Control program, and the loading worked reliably under all Data Link error conditions.

5.4 Updating and Debugging

While the Transmitter Site Control Program was being developed at Salisbury, which took some 15 months, the author had exclusive use of the PDP11 Backup computer as a host system, and down-line loading, at 9600 bits/s over a hard-wired link, took a few minutes only. There was a reasonably short turn-around time between runs of the program being developed and debugging was fairly simple, as it was always possible to produce an up-to-date assembly listing of any portion of the code that was giving trouble.

To help with debugging the changes to the Transmitter Control Program, a simple on-line debugging technique was included in the program. One of the Data Link commands, which are discussed in paragraphs 6.3 and 7.6, allowed for an operator at the Receiver Site to examine the contents of any location in the memory of the Transmitter Control Computer, and to alter them as required. However, the procedures to be followed at the Receiver Site were slow, and the facility had little use. ODT-11, the Paper Tape on-line debugging package, was also considered, but it was difficult to adapt to the Transmitter Control Program at the remote site.

During the initial installation, in September and October 1976, it became apparent that there were some errors in the Data Link communications protocol, because messages sent from either site often were repeated indefinitely. Because the Data Link handler code was similar in both the Receiver Site and Transmitter Site programs, it was possible in this case to make the necessary changes to the Transmitter Site program, and observe the result, entirely from the Receiver Site, without the programmer needing to visit the Transmitter Site at all. Some changes of a straightforward nature in other program areas were later also done from the Receiver Site, when there seemed to be little chance of trouble. In these cases coding details were sent from Salisbury to the Alice Springs field team, who carried out the necessary editing, assembly, loading and checking.

After the initial installation, three major updates were performed on the Transmitter Control Program, made necessary by equipment and operational changes during the development of the JINDALEE experiments. In each case, at least one month was spent in preparing the software at Salisbury, with the greatest care being taken with program design. Some algorithms could be developed in specially written RSX tasks, using the assembly language, and others were developed on the IBM370 Central Computer, emulating the PDP11 with a program running under TSO. With this technique, storage variables in a test program were given the names of PDP11 registers and program locations, and program statements were designed to emulate machine instructions one by one. The technique was quite successful, and the algorithms were error-free when copied into PDP11 assembly language.

Although the modified program was known to be free of language errors, it still had not been run in its entirety because the only computer with the hardware configuration needed was at the Transmitter Site. A magnetic tape copy of the program and a hard-copy of its assembly listing were then sent to the Receiver Site, and the author travelled to the Transmitter Site, taking another copy of the assembly listing with him. Debugging of the program then proceeded as follows. A colleague at the Receiver Site, usually from the field team, down-line loaded a newly edited and assembled version, and the author diagnosed any bugs by observing the program in action, requesting further editing etc. until the program was working correctly.

Sometimes the cause of a program error would be obvious, for example when a spelling mistake appeared in a printed message; but most bugs were far more subtle, often involving incorrect linkage to a subroutine, or the overwriting of data fields or executable instructions. The Transmitter Control Program was designed to halt whenever it encountered an illegal instruction or an address error. When this happened, core memory was examined by manipulating the switches on the Operator's Console, to study the system stack, program counter, general registers, and other information relating to the program at the moment it halted. By reference to the assembly listing produced at Salisbury it was then possible, in theory at least, to find the cause of the trouble. After several changes had been made by editing at the Receiver Site, the program would no longer be effectively described by the assembly listing. When this happened, the operator at the Receiver Site would be asked to obtain an assembly symbol table, as described in 5.2 above, and relay particular location values to the Transmitter Site to allow the listing to be referred once more to the actual program. However, it was generally possible to find a desired section of the currently loaded program by searching through core memory for an octal data pattern corresponding to some code in the original listing. As portions of faulty code were discovered they were noted for future editing correction, and in the meantime machine code patches were toggled into the computer's memory, using the switches on the Operator's Console, to allow other portions of the program to be tested. When enough editing changes had been worked out, or when the program had become so badly corrupted by accidental overwriting as to be no longer useable, details of the changes were sent to the Receiver Site. This might be done by using the program's keyboard communication facility, if that was still possible, otherwise the HF voice radio circuit between the two sites would be used. When editing and assembly were finished, the program would be reloaded and tried again.

The debugging procedures described above were continued until the program with the new features added had worked correctly with the rest of the JINDALEE system during normal operations. Each time a major program update was done, 4 or 5 serious coding mistakes were discovered during debugging at the Transmitter Site, and these generally took about 4 h each to track down and correct. With time being allowed for the operations at the Receiver Site, and for observation of the program when it was again integrated into the overall system, each visit to the Transmitter Site for program development involved some 40 or 50 h of duty at the site.

6. THE PROGRAM IN USE

6.1 Control Modes

An outline of the equipment control functions required at the Transmitter Site has been given in paragraph 2. Most of the relevant equipment was designed to be operated under either manual control or computer control, with a changeover switch for the operator to change between the two, and the computer program was designed to have two major states, for 'Local' and 'Data Link' control. Digital status lines were connected from the changeover switches to inform the program of the manual/computer control status, and the program could be changed from Local control to Data Link control by operator action.

The equipment control modes could be summarised as follows:

(a) an individual piece of equipment that was under Manual control was controlled from its knobs and switches, instead of from its external TTL (Transistor-Transistor Logic) control lines;

(b) under Local control, all equipment not under Manual control was controlled by TTL signals, which could be changed by the computer when the operator typed in commands at the DECwriter terminal; and

(c) under Data Link control, all equipment not under Manual control was controlled by TTL signals, which could be changed by the computer in response to messages sent over the Data Link from computer programs at the Receiver Site, while equipment status messages were sent back over the link to the Receiver Site.

The operator at the Transmitter Site could switch his program between Local and Data Link control at any time by typing CTRL/P, one of the ASCII keyboard control codes. If the program was under Local control when CTRL/P was typed, it switched immediately to Data Link control; but if it was under Data Link control, then additional input was requested to verify that the program should be switched back to Local control. This control mode switching was normally done by arrangement with the staff at the Receiver Site, as the current operating mode was of particular importance in the control of the Transmitter Site equipment.

The different control modes were designed for operational and safety reasons. It was believed that all of the computer-controlled equipment at the Transmitter Site ought, at any one time, to be under the exclusive control of either one site or the other, and that when a piece of equipment was switched to manual control, the computer should not be able to have any control over it at all. This last requirement, for manual control, was more difficult to achieve in the case of the HF power amplifiers, because each of them had several functions that could be exercised manually, and also because they were the most potentially hazardous equipment being controlled. An amplifier could be logically detached from computer control by use of one of the Local commands, which had the effect of removing it completely from interaction with the program, thus providing an extra level of isolation when it had to be worked on for maintenance. This is discussed further in paragraph 7.9.

6.2 Local Control

Under Local control, most of the control functions from the Receiver Site were locked out, and the Transmitter Site operator had full control of the equipment. The most important Local control operations were the assignment and detaching of HF power amplifiers, and the running of various system and equipment tests. It was also possible for the Transmitter Site as a whole to be operated under Local control as part of the overall JINDALEE system, with parameter values being communicated over the HF voice link or by some other means, but this was only done for special experiments.

To enable the Transmitter Site operator to carry out effective Local control, every effort was made to provide him with a conversational keyboard procedure that would prompt him for the values and settings needed, and provide meaningful diagnostics when mistakes were made. Equipment and other parameter settings were changed by typing-in decimal numbers through the DECwriter terminal. The first number needed was the Command Number, to identify the type of function to be performed. As an aid to the operator, a list of the command numbers and their corresponding functions was posted up on an equipment panel facing the DECwriter. A similar list is shown in Appendix II. When the command number had been accepted, the program prompted the operator for the parameters, with a short description of each parameter and the allowed range of its values. If several numbers separated by commas were typed on one line, then operator prompting messages would be suppressed. An incorrect value would result in a diagnostic message, and the parameter would be requested again. Input of the command could be abandoned at any time by typing CTRL/Z, or by allowing the keyboard to time out with more than 15 s between keystrokes. When all the parameters had been entered, the program printed a complete listing of the function and its parameter values, and asked the operator to confirm that this indeed described the operation intended. If the reply was unrecognizable, the question was repeated; if it was 'N' for no, or CTRL/Z, or if the keyboard was allowed to time out from lack of operator response, then the command input was abandoned; but if the reply was 'Y' for yes, then the command was executed immediately. Some examples of Local keyboard control are given in Appendix IV.

6.3 Data Link control

For exercising Data Link control from the Receiver Site, use was made of Receiver Site communications tasks that handled messages to and from the Transmitter Site, in the same way as they handled messages internally between other Receiver Site tasks.

Each Data Link command was encoded in ASCII characters, and was made up of a decimal command number, followed by the parameter values or other data. The command numbers are listed in Appendix II. An example of a Data Link command would be the character string '38,150, 12, 30, 0', which is Command 38, used for setting the time at the Transmitter Site, with the parameter values for a time of 150 days, 12 h, 30 min, and 0 s. Most of the Data Link commands were generated within user programs and dispatched to the Transmitter Site by the use of calls to standard subroutines. Their subsequent decoding within the Transmitter Control Program followed the procedures also used for decoding Local commands, except that if an error was detected, the rejection of the command was reported back to the Receiver Site. A correctly decoded Data Link command was executed immediately by the Transmitter Control Program.

Messages sent back from the Transmitter Site were received in their destination tasks by the use of subroutine calls to the communications system.

6.4 Message logging

The Transmitter Control Program maintained a printed log of all significant changes in operating conditions at the site, with each entry identified by the time at which the event occurred. The information recorded included changes in operating parameters caused by Local and Data Link commands, changes in equipment status such as manual control and fault conditions, and operator communications between the sites.

Some operations at the site, especially the starting of the radar and the running of the backscatter sounder, could generate a great deal of printed output, because of the repetitive nature of the commands and HF power amplifier operations. A "short" message format was provided for these, to print out only a summary of the parameters set up when a sub-system started operating, while the "long" format was normally used for system fault analysis, when every last detail might be needed. The message format was chosen with one of the system commands, from either Local or Data Link control.

6.5 Operator communication and other facilities

The operators at the two sites could send each other plain language messages by typing a letter and a slash, followed by the message text; for example, the Transmitter Site operator could send a message to his colleague at the radar console by typing 'R/MESSAGE...'. The operators could also attract attention at the other site with "alerts" with 3 levels of urgency - level 1 for "Low" level set off the BELL character in the terminal at the other end, level 2 for "Medium" level set off some Sonalerts to give a somewhat louder beep, and level 3 for "High" level rang a large alarm bell. At the Transmitter Site the alert was sent by the typing of the letter 'A' followed by the level required, for example A2 for the Sonalerts.

The Transmitter Site operator had other keyboard facilities as well; for example, he could get a complete listing of all parameters and equipment settings by typing S/; and he could reset all DECwriter and Data Link I/O operations by typing CTRL/Y.

7. PROGRAMMING TECHNIQUES

7.1 Overview

The program was made up of a main routine and 9 interrupt and trap service routines, with a number of utility subroutines, data tables and flags shared between them. Many of the utility subroutines, used for numerical conversion etc., were coded re-entrantly, so that a single copy of a subroutine in storage could be executed by different routines virtually simultaneously without mutual interference. Data linkage between the different sections of the program was maintained by tables and flags, updated dynamically to reflect equipment settings and status, operating modes, I/O queues, radar and sounder parameters, etc. I/O queues for the DECwriter console and the Data Link were maintained by calls to subroutines which raised the processor priority high enough to prevent interrupts while the queue pointers were being altered, and restored the former priority when the operation was finished.

7.2 Start and restart

The Transmitter Control Program could be started initially from switch settings on the Operator's Console, or it could be started by the Absolute Loader when down-line loading was complete. On startup, the system stack pointer was set and various table entries were initialised. The interrupt-serviced devices, viz. the DECwriter, the Data Link, the timing card, the A/D converter and the mains clock, were all made ready by setting the Interrupt Enable bits in their Control and Status registers. An introductory message and a list of equipment status and operating parameters was then printed, and a message was sent over the Data Link to the Receiver Site to ask for a Data Link command to be sent back with the current time of day, to allow approximate synchronisation of the two sites within some tens of milliseconds. The main program was then entered.

The main program consisted of a loop which was repeated indefinitely; within the loop the program printed operator messages as they appeared in the printer output queue, and it processed all keyboard input, including Local command decoding and inter-site operator communication. All other program operations were done by servicing processor interrupts from the devices mentioned above.

Recovery from power failure followed a procedure similar to restarting. Rather than try to restore the program environment, i.e. the program counter and general registers etc. as they were at the time of the power interruption, the program was written to reset the system stack and jump to the initial-start section again. It was considered that a power failure caused so much damage to the operating environment, causing, in particular, loss of timing and power amplifier settings, that the system needed to be almost completely re-initialised.

7.3 Keyboard and Printer handling

The keyboard handler comprised an interrupt service routine that stored input characters in a circular read-ahead buffer, along with a keyboard input subroutine called from the main program loop, that removed the characters from the buffer to process them. Most of the standard keyboard facilities found in the simpler RSX terminal handlers were supported; for example, typing RUBOUT deleted the current character, CTRL/U deleted the current line, and CTRL/R repeated the current line. The typing of CTRL/Z set the processor C bit on return from the subroutine, and a keyboard time-out set the V bit. Time-out was handled in conjunction with the timing card interrupt service routine, which once a second decremented a count that was later tested by the keyboard input subroutine.

When the operator wished to change between Local and Data Link control he typed CTRL/P; if the Transmitter Control Program was operating in Local control mode, the keyboard input subroutine then printed a prompting message, and called itself recursively to obtain the reply as to whether or not Data Link control was required. Typing CTRL/Y caused the keyboard interrupt service routine to re-initialise the queue pointers for the keyboard, printer and Data Link handlers, and to reset the Data Link protocol.

Printer output either was handled immediately, for keyboard echo and operator prompting during Local control dialogue, or was obtained from queued messages that were processed whenever the keyboard was not in use. The text of a queued message was preceded by a 9-character time code, representing the day, hour, minute and second at which the message was generated. The time was maintained by the Timing Card interrupt service routine. When the message was eventually printed, the time code was expanded into a suitable text string and printed left-justified on the page, while the rest of the text was indented a few places to make the times on the message log stand out.

7.4 Data Link protocol

The Data Link Protocol was the procedure designed for passing messages in both directions over the Data Link. It was designed by Mr J.C. Mackenzie, formerly of Cybernetic Electronics Group, DRCS.

The protocol was designed so that messages corrupted by transmission errors should not be allowed to pass, and no message should be lost in transit. Each message was allocated a sequence number using modulo-32 arithmetic, with one group of 32 sequence numbers used for transmission from the Receiver Site, and another group for transmission from the Transmitter Site. A message block was made up of a header, the message text, a trailer, and a checksum. The block header contained a control character and the message sequence number, and the trailer comprised other control characters and sequence numbers used by the protocol for sending back acknowledgements etc. The checksum was the two's complement of the arithmetic sum of all the preceding bytes in the block. In addition to checksum checking by the program, the DL11 receiver interface checked each character for parity errors, using a parity bit inserted by the DL11 transmitter interface at the other site, and any error was reported. A message that passed all the validity tests was acknowledged by sending back its sequence number and the ASCII 'ACK' code in a message trailer; but any messages rejected by logic or syntax tests were "acknowledged" by the ASCII 'NAK' code, and the Data Link handler at the other site then reported a possible programming error. Messages corrupted in transmission, causing parity or checksum errors, were not acknowledged at all and were thus eventually re-transmitted.

The link driver in the Transmitter Control Program was similar to the one at the Receiver Site. It comprised two interrupt service routines, one for the DL11 transmitter and the other for the DL11 receiver, closely linked through tables of message sequence numbers and flags indicating the progress of messages being transmitted and received. A message to be sent to the Receiver Site was queued by a standard subroutine call which added the text address to a circular buffer, so that when the Data Link handler was ready it took the address from the buffer and processed the message. Every message received from the Data Link was processed by Data Link command decoding, to be described later.

7.5 Numeric operations

The principal numeric formats used in the Transmitter Control program were 16-bit and 32-bit fixed point binary for arithmetic operations, and packed BCD (Binary Coded Decimal) for equipment I/O, which fitted two decimal digits into each byte. A precision of 32 bits for binary data satisfied the 24-bit requirement mentioned in paragraph 4, while making use of the PDP11 instruction set for extended-precision fixed point arithmetic. Subroutines were written for converting between the binary and packed BCD formats, using an 8-character ASCII decimal string for intermediate results.

A numeric input conversion subroutine was written, to handle character strings in any of the FORTRAN I, F, E or D formats. The output was 32-bit binary. As a basis for command decoding and for generating error messages, the subroutine returned a flag to show whether the number had finished with the end of the record or with a comma, and it provided a pointer to the first syntax error encountered.

Other conversion subroutines produced ASCII decimal and octal strings for output to the printer or the Data Link.

Addition and subtraction of numbers used straightforward machine instructions, but multiplication and division had to be done with specially written subroutines, because the basic PDP11/10 does not support these operations. Multiplication or division by an integer power of 10 was done by converting the number to an ASCII string, and then converting back again with an address pointer moved along by the required number of bytes, where each byte represented one decimal place; but other cases were handled by add, subtract and shift instructions.

Subroutines were written to perform a fixed-point cartesian to polar conversion, required for the phase and amplitude measurements to be described in paragraph 7.12.2. The subroutines computed the logarithm of a complex number (X+iY), where X and Y were fixed point numbers. The real part of the result was returned as 10*log(X*X+Y*Y), in units of decibels; and the imaginary part was returned as arctan(Y/X), in units of degrees. The calculations used the basic arithmetic operations described above, with table lookup and linear interpolation. The precision of the result was 0.1 dB for the real part, and 0.1 degree for the imaginary part.

7.6 Command decoding and execution

Command inputs from both the local keyboard and the Data Link were interpreted through an extensive command table, which took up about 25%

of the computer's available storage. There were 38 possible commands, arranged in a chained list. Appendix II is a list of command numbers with the purpose of each command, while the structure of each entry in the command table is shown in Appendix III.

Each entry started with a byte containing the command number, followed by a byte with a bit mask to define the operating conditions for the command, and a word containing the address of the next entry in the table. The bit mask included bits to tell the program whether the command could be used with Local control or Data Link control or both; whether command logging should be suppressed when the "short" message format was in effect; whether command logging should be handled by the standard subroutines or by special code; and whether the command could be used when the radar equipment was set up in the Backup mode, which will be discussed shortly.

This header was followed by a list of parameters for the command, each with a permitted range of values, and address pointers to the text describing the command and its parameters. The parameter list was terminated by a zero, and this was followed by the text of the subroutine to be executed in response to the command.

Local commands were decoded in the main program. When the program was in the Local control mode, any keyboard input that was neither a request for inter-site operator communication, nor the characters S/ (requesting a system status report), was examined to see if it was a valid numeric system command. The first character field in the input string was converted to a binary number, and, if the conversion was error-free, the command table was searched for a matching command number. If the numeric input had a mistake in it, or if the command number could not be found in the table, or if the bit mask showed that the command was not intended for Local input, then the program printed " ?? [BAD COMMAND NUMBER]", and returned to its loop.

Another diagnostic message was printed if the command being asked for was valid, but was not appropriate for the current operating mode of Normal or Backup operation. In Normal mode, sounder and radar equipment were used for their expected functions, while in Backup mode the radar equipment was used for sounding or telecommanding and radar experiments could not be run at all, so that radar commands were not then applicable. Backup operation was discussed in paragraph 2.4. The program was switched between Normal and Backup modes by one of the numeric system commands.

Some examples of Local command input are given in Appendix IV.

Once the command number had been accepted, the operator proceeded to supply parameter values, as described in paragraph 6.2, with the command table supplying the descriptive texts and the allowable ranges of values. The numeric input subroutine, described in paragraph 7.5, provided a flag to show whether numbers were input several to a line and separated by commas, or were input only one to a line, and provided a pointer used by the program to show the operator any mistakes in numeric input. The values being input were stored in a table for later listing and use. When the operator finally typed 'Y' to execute the command, the main program entered the control subroutine in the command table with a 'JSR PC,(R1)' machine instruction, with general register R1 holding the subroutine entry address obtained from the table. In this command subroutine, the parameter values stored away during input of the command were put into use, for example by being converted to a packed BCD field for driving a frequency synthesiser, or by setting a software flag to request some change of state in the program. On return from the subroutine, the program branched back into its main program loop.

There were some Local commands, used for system testing, that suspended normal program operations while the program went into a loop. For example, Command no. 6, listed in Appendix II, allowed the radar waveform generator to be set up as a programmable signal source. For most of these special commands, normal operator communications with the Receiver Site were maintained with the DECwriter and Data Link handlers continuing to run. When any of these commands was invoked, the program printed out full instructions for the operator; normal program operation could be resumed by typing a particular control character.

All messages received over the Data Link were decoded by a subroutine in the DL11 receive interrupt service routine, as part of the Data Link handler. These included the keyboard messages typed in by the Receiver Site operators, as well as numeric commands generated by various RSX control tasks. The decoding algorithm was similar to that used for Local command decoding, but was simpler because the text of each command was in a single record, and there was no need for operator interaction. Detection of any command syntax error resulted in the NAK character being sent back to the Receiver Site with the message's internal sequence number, as discussed in paragraph 7.4, while a syntactically correct message caused the command subroutine to be executed, and a logging text was queued for the DECwriter printer. Any conflicts between Normal and Backup operations and commands were reported back to the Receiver Site with a suitable message.

7.7 Timing card

The timing card served several functions associated with radar waveform generation, and the computer could load its device registers with numbers for setting up repetition rates and delays for synchronising the radar at the Receiver and Transmitter Sites. The registers were loaded in response to standard system commands sent over the Data Link, or they could be set under Local keyboard control. The radar waveform equipment, which was supplied with some of its control signals by the timing card, comprised frequency synthesisers and their drivers, frequency translators, and other radio frequency equipment.

The timing card also had a Programmable Clock, driven from the station's frequency standard, which caused a program interrupt every 1 second precisely. The 1-second interrupt service routine had the following functions:

(a) it updated the station time of day, as used for message logging, the synchronising of operations, etc.;

(b) it kept "time out" counts for other routines, e.g. the keyboard handler, and the station alerts, which sounded for only a few seconds;

(c) it read the equipment status bytes, and generated printer messages to report changes of status, and when under Data Link control, it also generated status messages for the Receiver Site; and

(d) it supervised HF power amplifier operations for both the radar and the backscatter sounder.

The operation and control of the HF power amplifiers will be dealt with in paragraph 7.9.

7.8 Output to equipment

One of the program's principal data tables was a copy of the control bytes, of which there were 37 (decimal), for output to the equipment. A mains clock interrupt service routine copied the contents of this table 50 times a second to the output modules to set up the TTL output lines. Whenever any routine in the program moved a data pattern to this table, then the TTL control lines to the corresponding equipment would be changed within 20 ms, which was usually quite fast enough. If a faster response was needed, the table could be copied out within about 50 μs by use of the 'TRAP' machine instruction, which was handled by the same service routine but at a higher priority.

7.9 HF power amplifier control

Figure 2 is a diagram of the various control functions that were needed for HF power amplifier operation.

Each of the 5 HF power amplifiers had associated with it 4 digital control lines driven from the computer via the asynchronous byte interface, and 7 digital status lines going back to the computer. These lines were grouped into 4 output bytes feeding signals to the amplifiers and their control equipment, and 7 input bytes accepting status information back from the amplifiers and associated equipment. Each byte comprised 8 bits; of these, the 5 least significant were distributed each to a particular amplifier, with the least significant bit going to Amplifier no. 1, the next least significant bit going to Amplifier no. 2, and so on.

One aspect of HF power amplifier control that proved valuable in practice, was the ability to associate logical unit numbers, in the range 0 to 5, with the physical amplifiers. The allocation was done with Command no. 7, one of the Local keyboard commands in Appendix II. The default allocations, from down-line loading, were for logical units 1 to 4, the radar units, to correspond to physical units 1 to 4, and logical unit no. 5, the sounder unit, to correspond to amplifier no. 5. Whenever an amplifier had to be taken out of service, its physical number would be allocated to logical unit no. 0. Logical bit masks were used in the program for selecting and controlling combinations of the 5 amplifiers, with the 5 least significant bits being set or cleared according to the control needed over the corresponding physical (as opposed to logical) amplifier.

The control lines from the computer to each amplifier were:

(a) a Tune Initiate line, which needed to be asserted for long enough for an electro-mechanical relay latch to operate, starting the tuning servos so that the amplifier could tune to the input frequency and load into its antenna system;

(b) a Keying line, to bias the amplifier on or off; when keyed on it could radiate, keyed off it could not;

(c) a Continuous Sweep line, to put the tuning servos into a state where they could fine-tune and load continuously, as needed for sounding operations; and

(d) an EHT Trip line, which when asserted would turn off the amplifier's Extra High Tension of several kilovolts, as a safety measure.

The status lines coming back from each amplifier were:

(a) an EHT Ready line, set high when the Extra High Tension was available, which occurred whenever the vacuum tube heaters had warmed up, all safety interlocks were closed, and the EHT had been turned on by the operator;

(b) a Tuning Complete line, set high when the amplifier's electro-mechanical tuning servos had successfully finished tuning, and had loaded into the antenna system;

(c) a Tuning Fault line, set high when the tuning servos could not complete tuning within the time allowed, which had been adjusted to 25 s;

(d) an Overload line, set high when the plate current from the output stage became excessive;

(e) a High VSWR line, set high when the Voltage Standing Wave Ratio of the amplifier's load, as indicated by the reflected power dissipated in the output stage, was too high;

(f) a Manual Keying line, set high when the amplifier was switched to Manual keying; and

(g) a Manual Continuous Sweep line, set high when that function was switched to Manual control.

The sinusoidal HF signals to be amplified for output to the antennas were obtained from various programmable frequency synthesisers, feeding frequency translation equipment. In the case of the radar, the drive signal to the amplifiers passed through a programmable attenuator, to set the output power level, and then through a programmable delay line system used for beam steering. Details of the radar signal format, and the equipment used for generating it, are beyond the scope of this paper; but some details of the the sounder signal are given later in paragraph 7.11.

The high power outputs from the radar amplifiers were fed through 4-way power splitters to drive the antenna subarrays, as described in paragraph 2.1. Antenna faults could cause considerable power dissipation and temperature rise within the splitters, for which over-temperature sensors were provided. These were in addition to temperature switches that turned on water cooling to the splitters under normal circumstances. The over-temperature switches were connected into the safety interlock system, and provided 4 status lines to the computer, in addition to those already mentioned, grouped into one byte.

All HF power amplifier operations were controlled by the timing card interrupt service routine, using software flags to control the sequence of operations within the routine. An amplifier was physically ready for tuning when any fault conditions had been cleared and its EHT was available. If tuning was required, the program then set the source of low-level HF drive to the required frequency, and the signal level was set to that needed by the tuning servos. It set the Tune Initiate line high for 1 second and waited for either the Tuning Complete status line to report success, or the Tuning Fault line to report failure.

Once a tuning cycle was started it could not be interrupted, being under the control of electro-mechanical devices within the amplifier. If system control again requested tuning before a tuning cycle was finished, then the command subroutine that included the code for setting the HF drive and initiating the tuning was not executed in full. Instead, it stored away the entry address to the equipment control part of the subroutine, and that part of the code was later executed by the timing card interrupt service routine when the tuning cycle was complete. The list of command subroutine entry addresses was kept in a set of circular buffers, and was examined every second by the timing card routine.

When tuning was complete, the program put the amplifier into service according to the parameters fed in by system commands. For the radar this involved setting up the HF modulation and on/off keying, and the drive level and beam steering; for the sounder, it involved setting the continuous sweep control or the telecommand modulation, and on/off keying.

If a tuning fault was detected in an amplifier, the program tripped off the EHT supply to that amplifier to prevent damage to the tuning circuits caused by re-cycling of the servos. In this and all other HF power amplifier fault conditions, the program keyed the faulty amplifier off, and then removed it from control, by altering the references to it in the 5-bit logical masks mentioned earlier in this paragraph. The amplifier was again put back under program control when the faults were cleared, which had to be by manual intervention. The program regarded an amplifier as faulty if its EHT was not ready, so that any amplifier could be removed from service by having its power turned off; but before it could be worked on, it had to be logically detached by Local command.

7.10 Radar control

When the program was asked to implement a new set of radar parameters, it checked the frequency limits of the proposed signal against a table of Prohibited Frequency bands, which included the frequency allocations of important communications services that could perhaps be disrupted if the JINDALEE radar were operated in the same band. An identical Prohibited Frequency table was used by the radar control programs at the Receiver Site when radar parameters were chosen; checking at the Transmitter Site was included as an additional safeguard. Each table entry at the Transmitter Site consisted of the lower and upper limits of a prohibited band, in kHz, stored in the form of 1-word integers. The Transmitter Control Program had two copies of the Prohibited Frequency table, both down-line loaded from the Receiver Site into fixed locations at the top end of core storage. In case the table should need to be corrected, one copy at the Transmitter Site could be examined and altered using Local keyboard commands, while the other could be altered with a Data Link command from the Receiver Site. However, changes to the tables were normally edited in at the Receiver Site, where they were kept in a MACRO-11 source module, and were assembled and down-line loaded to the Transmitter Site.

When radar operation was requested by the execution of the command subroutine that set up the radar frequency, the Transmitter Control Program calculated the relevant frequency limits of the radar, and the fixed frequency at which the amplifiers should be tuned. It then checked the tables of prohibited frequencies, making sure that the number representing the lower limit of each band was less than the number for the upper limit, and that the two copies of the table were identical. When the table had passed these tests, the radar limits were checked against every entry in the table, to make sure that no part of the radar signal should overlap any prohibited band. If the program found any error in the tables or any attempted violation of a prohibited band, then appropriate messages were generated, to be sent back to the Receiver Site and printed locally. This program prevented the radar from operating if the Prohibited Frequency tables were wrongly formatted, or if the radar parameters resulted in frequency limits inside Prohibited Frequency bands. The checking at the Transmitter Site did not hinder the operation of the radar, but proved useful in locating any errors in the radar frequency selection at the Receiver Site.

The program was written so that when the radar was operating, the radar equipment had to be either all under computer control, or else all under Manual control. If any of the radar equipment was switched to Manual control, then the program set output control bytes to shut the radar down temporarily by every means available - by keying off, by setting the amplifier drive level as low as possible by means of the programmable attenuator, and by setting the frequency to a value outside the amplifier passband. If the operator wanted the radar to work with its equipment under manual control, then he had to manually over-ride the above emergency settings. The computer restored the radar control bytes when it again had control of all the radar equipment.

7.11 Sounder control

The sounder sweep frequency limits were set under program control, and for the purpose of synchronising the Transmitter and Receiver Site sweeps, resetting to the bottom of the sweep could be commanded for a specific time. The sweep-frequency controller used for Normal operation was free-running and independent of the computer, in that it continued sweeping upwards at 33.3 kHz/s, and resetting to the bottom frequency whenever it reached the top of the sweep. The controller informed the computer of sweep resets by the setting of a bit in one of the status bytes. In Backup mode, the Transmitter Control program had explicit control of certain aspects of the radar waveform control equipment, which then caused its output frequency to sweep upwards at 33.3 kHz/s. The radar equipment was set to sweep over a 100 kHz segment with a repetition period of 3 s, and a frequency synthesiser driving the output frequency translators jumped by 100 kHz after each 3 s sweep. This had the effect of a continuous sweep, except that every 3 s there was a waveform discontinuity with a duration of about 100 μs, while the program updated the synthesiser settings.

As explained in paragraph 2.2, the HF power amplifier to be used for sounding had to be pre-tuned to a fixed frequency at the bottom of a segment of the sweep, before it could be keyed on and switched to continuous sweeping operation. This was necessary whenever the sounder system, either Normal or Backup, was commanded from 'disabled' to 'enabled'; when it was switched from Telecommand to Sounder operation; and when the sweep controller reset the frequency to the bottom of the sweep range. The program allowed a fixed time of 18 s for tuning, and when actual tuning took less than 18 s, which was usual, the amplifier remained idle for the remaining time. The sweep frequency advanced by 600 kHz in 18 s, unless there had been a sweep reset in the meantime; the program kept track of the frequency 'now' and the frequency expected in

18 s time, with due regard for anticipated sweep resets. Whenever tuning was needed, the drive was set to the required fixed frequency (while the sweep controller continued cycling), then Tune Initiate was asserted, and after 18 s if tuning was complete and there had been no faults, the amplifier was keyed on and switched to Continuous Sweep. Sounder data gathering could then be resumed at the Receiver Site.

Damage could be done to the tuning circuits during sweeping if the drive were interrupted momentarily, which sometimes happened because of a fault in the sounder frequency synthesiser. When this happened the servos started hunting, and if they could not achieve fine tuning and loading the tuning circuits and switches might be burnt out. A 'Sounder Drive Failure' circuit was built to detect drive dropouts for Normal sounder operation and report them to the program through a status byte. The program response to a dropout was to key the amplifier off, and as soon as the operator reset the Drive Failure circuit, the program tuned the amplifier and keyed it on again.

7.12 A/D data gathering experiments

The A/D converter has been mentioned in paragraph 3.1. It had 8 analogue input channels of 12-bit precision, handling voltages in the range -10 to +10 V, at a sampling rate of 20 samples/s precisely. Each sample value was stored in the 12 most significant bits of a word, and the channel number, 0 to 7, in the 3 least significant bits. The sample words were buffered in a 32-word FIFO (first in, first out) buffer within the A/D interface; the DR11-C module caused a processor interrupt when the buffer was about half full. The A/D interrupt service routine then read in 16 samples through one of the registers of the DR11-C interface, and dealt with them according to their channel numbers.

Three experiments were handled, viz. Ground Conductivity monitoring, Transmitter Phase and Amplitude monitoring, and Data Link Signal Strength monitoring. The data gathered for each was pre-processed by the Transmitter Control Program, and queued for output via the Data Link or the DECwriter console, depending on the current control mode. If the Transmitter Control Program was under Data Link control, then at the Receiver Site the data was further processed for recording and display.

7.12.1 Ground Conductivity Monitoring

The Ground Conductivity equipment used a simple Wenner conductivity monitor, comprising four sets of probes driven into the ground. Each probe set had four probes in line; an audio frequency current passing between the outer probes was measured, the potential difference between the inner probes was also measured, and from the readings the conductivity could later be estimated. The readings were normally requested by a Data Link command. The experiment was started by a TTL signal sent by the computer to the equipment in the field, where a mechanical sampling switch was set going. Calibration levels and actual readings were taken in sequence and converted to DC levels, which were sent back as a voltage histogram to one channel of the A/D converter. The A/D interrupt service routine demultiplexed the histogram, assembled the readings into messages, and queued them for output. A number of tests were built into the program to check the histogram coming in from the field, because the sampling switch tended to operate erratically and the set of readings was sometimes not completed. An error message was queued when appropriate.

7.12.2 Transmitter Phase and Amplitude Measurements

The Transmitter Phase and Amplitude measurements could be requested by a numeric system command, either via the Data Link or input locally. Under Local control, they could be requested by typing an asterisk during the execution of the command that called up a fixed frequency from the radar generator. Under Data Link control the measurements were sent to the Receiver Site, where they were used for estimating the radiation pattern of the Transmitter Site radar antennas, while under Local control the results were printed as a list of phases in degrees and amplitudes in dBW. The diffraction theory calculations needed for the Receiver Site were far too complicated to be hand-coded into the Transmitter Control program; it only went as far as calculating the complex logarithm as described in paragraph 7.5.

During Transmitter Site equipment checking, the measurements proved very useful for doing radio frequency measurements that would have otherwise needed an analogue Vector Voltmeter. The results from this computer-aided technique were about 10 times more accurate than obtainable from the analogue instrument, and greater operator convenience allowed each measurement to be done some hundreds of times faster, within seconds instead of minutes.

When the measurements were to be done, radar and sounder operations were interrupted for about 0.8 s, and during this interval the frequency synthesisers, electronic switches and other equipment were put into a special configuration. The radar frequency was set to a suitable fixed value, and the sounder synthesiser to a frequency offset from it by 5 Hz. At the inputs to the radar 4-way high-power splitters there were co-axial bi-directional couplers whose forward power samples were mixed with the offset signal, using electronic mixers, to produce four 5 Hz sinusoidal signals, and each of these was fed to an input channel of the A/D converter. The A/D sampling rate of 20 times a second meant that the inputs were sampled exactly 4 times a cycle, producing successive cosine and sine values; these were averaged over 4 cycles of the 5 Hz waveform before the phases and amplitudes were calculated. The amplitudes were corrected to allow for the calibration of the bi-directional couplers, by means of tables of coupler loss versus frequency, using linear interpolation. All phases were referred to Amplifier no. 1. When the measurement was finished, the results were queued for output and the equipment was restored to its former configuration and settings.

7.12.3 Data Link Signal Strength Monitoring

The Data Link Signal Strength monitoring was done as part of a study of VHF propagation between the two sites. A DC output line from the Data Link receiver gave an indication of received signal strength, and this line was connected to a channel of the A/D converter. As the experiment called for only one sample in 10 s, every 200'th sample from the A/D was stored, and a message transmitted a batch of 12 readings, representing 2 min data. The data gathering went on continuously, but queuing of the results for output could be started or stopped by a system command.

8. DISCUSSION

8.1 Facilities

In an experimental project like JINDALEE, where much of the data gathering and processing are controlled with the help of digital computers, it is inevitable that the experimenters and others will want to modify the computer programs from time to time. The person responsible for changing a computer program in the field needs some basic tools for doing his work. The following list shows some of the requirements, and how they were met when the Transmitter Control Program had to be written and changed.

(a) The computer programming language should preferably be one that can be easily read and understood by the user of the program, who will not necessarily be a skilled programmer. In the present case only an assembly language was available, and programs in these languages are notoriously hard to follow, even when the program is written with adequate comments.

(b) A convenient means is needed for generating, storing and editing the files containing the source-language code. Because of limitations in the PDP11 Backup system this had to be done on the IBM370 Central Computer at Salisbury, while at Alice Springs it was done at the Receiver Site.

(c) A language translator is needed to convert the source code into object or machine code, and produce a program listing. Language translation at DRCS was divided between the PDP11 Backup computer for assembly and the IBM370 Central Computer for program listing, a rather unsatisfactory arrangement, while at Alice Springs it was done on one of the computers at the Receiver Site.

(d) It is highly desirable that there should be facilities to allow program modules from separate sources to be linked together into an executable program. No suitable facilities were available, so that the program had to be coded as a single module without the advantages of subroutine libraries etc.

(e) During program development, a computer is needed with the necessary hardware facilities to allow the program to be run. After field installation, the only computer with suitable facilities was the one at the Transmitter Site.

(f) A means is needed for loading the program into the computer. This was divided between the Receiver and Transmitter sites.

(g) Debugging aids are needed when the program does not work as expected. The aids available were most inadequate, as they comprised only the Operator's Console switches and lights on the computer, and a program listing that could not be updated as debugging proceeded.

In the light of experience gained from working with the Transmitter Site installation, it can be recommended that any computer installation to be used at a remote experimental site should be self-sufficient in software facilities, preferably using an operating system. It should be possible to use a high-level programming language like FORTRAN, PL/I, or Basic; and it should be possible to store and load programs, and carry out any other software maintenance, without the need for intensive support from other sites and facilities. For the Transmitter Site installation, suitable operating systems might have been either RT-11 or CAPS-11, which can offer all, or most, of the facilities mentioned. It is also highly desirable that programs intended for the remote site should be run at the base installation during updating, using a computer that duplicates, as nearly as possible, the hardware features of the remote system; this facility would also be useful for investigating program malfunctions reported from the field.

8.2 Program design and operation

The Transmitter Control Program was written specially for the application described, and it was developed to a large extent along with the equipment that it was intended to control. This procedure was satisfactory for all concerned, as it assured reasonable compromises between the needs of hardware and software design.

In the field, the program and the control equipment proved quite adequate for their tasks. The site operators seem to have had few difficulties with using the program, apart from minor problems during training and familiarisation.

There were occasional program failures that may have been caused either by software faults or by transient hardware faults. When a fault occurred, whatever its cause, the trouble could usually be cleared by restarting from the Operator's Console or by typing CTRL/Y to reset the I/O handlers, but sometimes down-line re-loading was needed. The operators were instructed not to persist with using the program if it went erratic, but to down-line load it again. On the few occasions when the computer stopped completely, a power supply or other electronic fault was found to be the cause.

Once the program was operating under Data Link control it was slaved to the Receiver Site. The operators could then generally ignore it until summoned by an operator alert from the Receiver Site, unless they needed to attend to any equipment, or wanted to read the message log with its parameter settings and other information.

9. CONCLUSION

The Transmitter Control Program was written from first principles in the assembly language of its PDP11/10 computer, which took about 15 man-months of programming effort. The program was developed along with much of the equipment that it was intended to control. In its 27 months in the field it gave little trouble, either from operators' procedures or from equipment control.

Some updating of the program had to be done after installation, due to operational and equipment changes, and changing the program then proved to be rather difficult because the facilities needed were spread between several different computer installations. It is recommended that when a computer is installed at a remote site, then any software maintenance should be allowed for by the provision of reasonable programming aids at the same site, and that the base installation should be able to deal with the specific requirements of the remote site.

10. ACKNOWLEDGEMENTS

The author wishes to acknowledge the help of Mr. J.C. Mackenzie, formerly of Cybernetic Electronics Group, Salisbury, in system programming; of Mr. A.D Massie of Jindalee Project Group, Salisbury, for help and advice with equipment control; and of Mr. D.N. Warren-Smith, formerly of the Jindalee Experimental Facility, Alice Springs, for help at the Receiver Site during the long hours of program debugging.

REFERENCES

No.	Author	Title
1	Digital Equipment Corporation	"PDP11/04,05,10,35,40,45 Processor Handbook", EB-05138-76
2	Digital Equipment Corporation	"PDP11 Peripherals Handbook", 1976, EB-05961-76
3	Digital Equipment Corporation	"IAS/RSX11 MACRO-11 Reference Manual", DEC-11-OIMRA-B-D

APPENDIX I

ABSOLUTE BINARY FORMATS

When the '.ENABL ABS' directive is used at the start of the source code, each record of binary output from the MACRO-11 assembler has the following format:

(a) Record length, 2 bytes, used by File Control Service routines; available to the user when the record is read using the RSX 'GET$' macro;

(b) Load address for machine code, 2 bytes;

(c) Machine code.

The Paper Tape Absolute Binary format, used for down-line loading, is as follows:

(a) '1' byte (block header);

(b) '0' byte (block header);

(c) Record length, 2 bytes;

(d) Load address for machine code, 2 bytes;

(e) Machine code;

(f) Checksum, 2's complement of arithmetic sum of all bytes in the block.

APPENDIX II

LIST OF COMMAND NUMBERS

The following list is included to give the reader some idea of the scope of control offered by the Transmitter Control Program.

(a) Commands input to the program, from keyboard and/or Data Link. An asterisk shows that the command could be input only from the Data Link.

COMMAND	FUNCTION
3	Data Link signal strength calibration
4	Data Link echo test
5	Data Link remote terminal
6	CW test of radar
7	Set up power amplifier assignments
8	Keyboard prohibited frequency input
9	List prohibited frequency bands
10	List disagreements between prohibited bands
11	Enable / Disable radar
12	Radar drive attenuation
13	Radar beam steering
14	Radar frequency
15	Radar sweep slope
16	Radar interference suppression
17	Radar sweep rate
18	Timing synchronisation
19 *	Message from Radar operator
20 *	Alert from Radar operator
21	Single radar sweeps for site synchronisation
31	Enable / Disable sounder and telecommand
32	Switch to sounder operation
33	Radar Backup mode switch
34	Sounder sweep limits
35	Reset Normal sounder
36	Reset Backup sounder
37	Telecommand control
38	Set station time
39 *	Request Transmitter Site status
40 *	Prohibited frequency band from Data Link
41 *	Message from Data Logger operator
42 *	Alert from Data Logger operator
43 *	Receiver Site alert status
44	Request Ground Conductivity measurement
56 *	Transmitter Site message format
80	Wide band sweep blanking
81	Request radar amplitude / phase measurements
83	Data Link signal strength monitor
89 *	Remote On-line debugging facility

(b) Commands sent back to the Receiver Site

COMMAND	FUNCTION
51	Message for Radar operator
52	Report wrong operating conditions to radar
53	Radar power amplifier operation
54	Radar manual control status
55	Report violation of prohibited frequency band
61	Message to Data Logger operator
62	Send alert to Receiver Site
63	Report Local / Data Link control
64	Report wrong operating conditions to Data Logger
65	Sounder equipment operating status
66	Sounder equipment manual control status
67	Transmitter Site alert status
68	Report power failure, request for time setting
69	Error in prohibited frequency tables
70	Ground conductivity results
82	Radar Amplitude / Phase results
84	Data Link signal strength readings
88	On-line debugging results

APPENDIX III

FORMAT OF COMMAND TABLE

The entry for each command had the format:

```
Word 0   Byte 0   Command number
         Byte 1   Bit 7 set - valid for Local I/P
                  Bit 6 set - valid for Data Link I/P (Radar)
                  Bit 5 set - valid for Data Link I/P (Data Logger)
                  Bit 4 set - valid for 'Normal' mode
                  Bit 3 set - valid for 'Backup' mode
                  Bit 2 set - can be I/P from Data Link
                              while under Local control
                  Bit 1 set - use special logging of command
                  Bit 0 set - suppress logging if 'Short'
                              message format

Word 1   Address of next entry, or 0 if no more entries
```

Then 1 block for each parameter in the command

(1) If numerical data

```
.BYTE  Decimal scale factor for input (P factor)
.BYTE  no. of bytes in stored limits (1, 2, or 4)
Lower limit (integer)
Upper limit (integer)
Address of descriptive text
```

(2) If multi-choice data

```
.WORD  -<N+1>  ; where N=no. of options
Address of main text
Address of text for option no. 1
Address of text for option no. 2
   etc.
```

(3) If a text-string (message) command

```
.WORD  -128.
Address of descriptive text
```

(4) After the last parameter

```
.WORD  0
<Executable code, entered by JSR PC,****>
```

APPENDIX IV

EXAMPLES OF KEYBOARD COMMAND INPUT

(a) Setting up the frequency limits of the Backscatter Sounder

```
 34
 SOUNDER LOW MHZ[6 - 30]?  10
 SOUNDER HIGH MHZ[6 - 30]?  200
 OUT OF RANGE
 SOUNDER HIGH MHZ[6 - 30]?  2)
                               * ERROR
 SOUNDER HIGH MHZ[6 - 30]?  20

 ** LOCAL COMMAND...
 SOUNDER LOW MHZ = 6
 SOUNDER HIGH MHZ = 20
 IS THIS OK?  TYPE Y OR N ...  N
[ABANDONED]


 34,10
 SOUNDER HIGH MHZ[6-30]? 29

 ** LOCAL COMMAND...
 SOUNDER LOW MHZ = 10
 SOUNDER HIGH MHZ = 29
 IS THIS OK?  TYPE Y OR N ...
[KBD TIME-OUT]
[ABANDONED]


 34,10,29

 ** LOCAL COMMAND...
 SOUNDER LOW MHZ = 10
 SOUNDER HIGH MHZ = 29
 IS THIS OK? TYPE Y OR N...  Y
123:12:20:26   ...ENTERED
```

(b) Setting operating mode to Backup

```
 3$
 ?? [BAD COMMAND NUMBER]
 33
 MODE   NORMAL[1]  BACKUP[2]... 2

 ** LOCAL COMMAND ...
 MODE> BACKUP
 IS THIS OK? TYPE Y OR N ...  Y
123:12:21:15   ...ENTERED
```

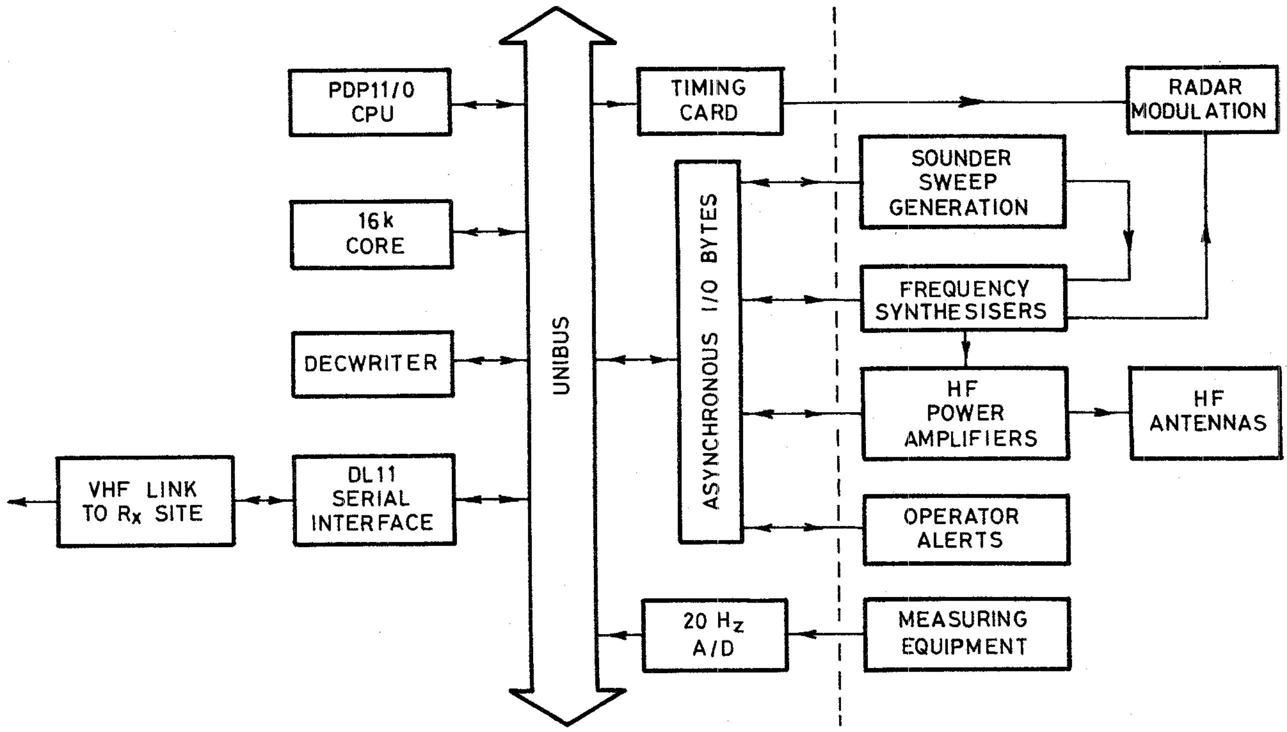

Figure 1. Computing hardware at transmitter site

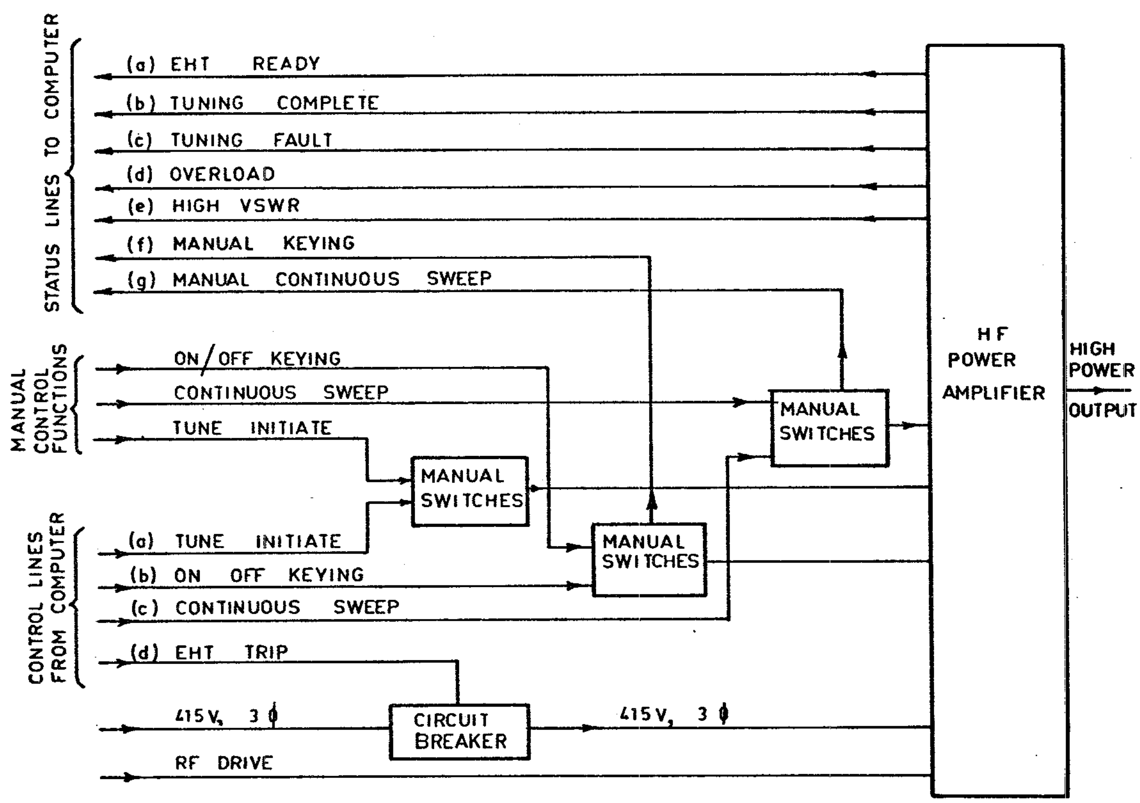

Figure 2. Control of an HF power amplifier

DOCUMENT CONTROL DATA SHEET

Security classification of this page: UNCLASSIFIED

1 DOCUMENT NUMBERS

AR Number:	AR-001-472
Report Number:	ERL-B123-TR
Other Numbers:	

2 SECURITY CLASSIFICATION

a. Complete Document:	UNCLASSIFIED ~~RESTRICTED~~
b. Title in Isolation:	UNCLASSIFIED
c. Summary in Isolation:	UNCLASSIFIED

3 TITLE

JINDALEE PAPER NO. 123

4 PERSONAL AUTHOR(S):

A.M. Forbes

5 DOCUMENT DATE:

February 1979

6

6.1 TOTAL NUMBER OF PAGES	38
6.2 NUMBER OF REFERENCES:	3

7 7.1 CORPORATE AUTHOR(S):

Electronics Research Laboratory

7.2 DOCUMENT SERIES AND NUMBER

Electronics Research Laboratory
B123-TR

8 REFERENCE NUMBERS

a. Task:	DEF 77/036
b. Sponsoring Agency:	DEF

9 COST CODE:

366965

10 IMPRINT (Publishing organisation)

Defence Research Centre Salisbury

11 COMPUTER PROGRAM(S)
(Title(s) and language(s))

12 RELEASE LIMITATIONS (of the document):

~~Distribution:~~ Approved for Public Release ~~sts for this document must be referred~~
~~to Chief Super~~ ~~Research Laboratory.~~

12.0	OVERSEAS	NO	1	P.R.		A	5	B	X	C	X	D	X	E	X

Security classification of this page: UNCLASSIFIED

Security classification of this page: UNCLASSIFIED

13	ANNOUNCEMENT LIMITATIONS (of the information on these pages):
	Sp ANNOUNCEABLE: to the Public ənts

14	DESCRIPTORS:		15	COSATI CODES:
a. EJC Thesaurus Terms	Computer programs Computer programming Computer systems programming Computer control Process control	Interactive programs Transmitters Data links Assembly language		
b. Non-Thesaurus Terms				

16	LIBRARY LOCATION CODES (for libraries listed in the distribution):

17	SUMMARY OR ABSTRACT: (if this is security classified, the announcement of this report will be similarly classified)

Security classification of this page: UNCLASSIFIED